THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 58, No. 7, September 1979
Printed in U.S.A.

An ADPCM Approach to Reduce the Bit Rate of μ-Law Encoded Speech

By H. W. ADELMANN, Y. C. CHING, and B. GOTZ

(Manuscript received March 14, 1979)

A new ROM-oriented ADPCM architecture for processing μ-law encoded speech is presented. The architecture is shown to offer: (i) simple hardware realization, (ii) flexibility of algorithm modification, (iii) fixed hardware complexity with respect to numerous nonlinear processings, and (iv) excellent time-sharing capability. Performance measurements of a hardware implementation are also included.

I. INTRODUCTION

Bit rate reduction from the present 64 kb/s per trunk is of interest both to reduce per-trunk cost of existing and planned digital transmission facilities and to prove-in digital transmission.

Adaptive differential PCM (ADPCM) coding of the type reported in Refs. 1 through 4 appears promising for bit rate reduction of *speech* signals. Our specific interest is the bit rate reduction of the conventional μ-255, 64-kb/s signal. The purpose of this paper is to present an ADPCM architecture that offers: (*i*) simple hardware realization, (*ii*) flexible algorithm modification, (*iii*) fixed hardware complexity with respect to numerous nonlinear processings, and (*iv*) time-sharing capability over many trunks. The performance of a hardware realization of such a coder, which is shared by 24 voice trunks, is also reported.

II. THE PROPOSED ADPCM ARCHITECTURE

2.1 Basic structure

DPCM with adaptive backward quantization and fixed one-tap predictor, as proposed and analyzed in Refs. 1, 2, and 3, is our point of departure. Figure 1 is the conventional block diagram of an ADPCM codec and its interface to a μ-law environment. The quantizer with scale (e.g., uniform step size) Δ is denoted by Q_Δ, while the "inverse

quantizer" which maps a quantization interval code IQ into a quantized signal level is denoted by Q_Δ^{-1}. The quantization scale $\Delta(i)$ is determined by a scale adaptation algorithm from the quantization index sequence $IQ(i)^*$, which is sent over the channel. We add a prime to the decoder variables.

Prior research on ADPCM assumes a fixed linear signal representation. In the case of the μ-law signal environment, this requires μ-to-linear (μ/L) and L/μ conversions, as shown in Fig. 1. The 14 bits per sample required by the linear representation is an indication of the complexity of the straightforward approach.

Our basic approach is to put as much of the ADPCM algorithm as is presently practical into read-only memory (ROM). This allows us to confine the 14-bit linear representation to the ROM table computation (i.e., to the firmware). We motivate our architecture by relating it to the more familiar architecture of Fig. 1. For this purpose, assume a uniform B-bit quantizer which can be considered to be a cascade of an infinite amplitude quantizer and a B-bit limiter. The argument to follow could be generalized to a nonuniform quantizer. (Note that the limiting is on the quantizer output code as opposed to input amplitude.) Retaining the limiter in its place but shifting the quantizer and inverse quantizer past the difference and summing nodes† produces Fig. 2 from Fig. 1. (For convenience, only a local decoder is shown together with the new notation.) To complete the transition to the proposed architecture as shown in Fig. 3, we first combine μ/L and Q_Δ into one block for $\mu/$APCM conversion via a ROM realization. We next note that, instead of $\hat{X}_L$, we could store any other absolute representation of the local decoder estimate; in particular, we use the μ-law representation. Combining Q_Δ^{-1} and L/μ into an APCM/μ block and combining the μ/L, predictor multiplication and Q_Δ into a $(\mu/\text{APCM})_A$ block, we obtain the proposed architecture shown in Fig. 3 where $\mu/$APCM, APCM/μ, $(\mu/\text{APCM})_A$, and SCALE ADAPTATION are to be implemented with ROMs.

The effects of these actions are discussed below. By moving the adaptive quantizers through the difference node, we can combine many nonlinear functions, which are difficult to realize in combinatorial logic, into ROM tables. The generation of these tables can be achieved with an off-line computer, where the 14-bit precision inherent in the μ-law code is used. In addition, the outputs of these ROMs can be restricted to 8-bit APCM words that can be easily manipulated. We next examine in more detail our proposed quantization and scale adaptation scheme.

---

* $IQ \epsilon \{\pm 0, \pm 1, \cdots, \pm 2^{B-1} - 1\}$, where B is the number of bits/sample sent over the channel.

† Note that $Q(x + y) \simeq Q(x) + Q(y)$ to within one quantizer interval.

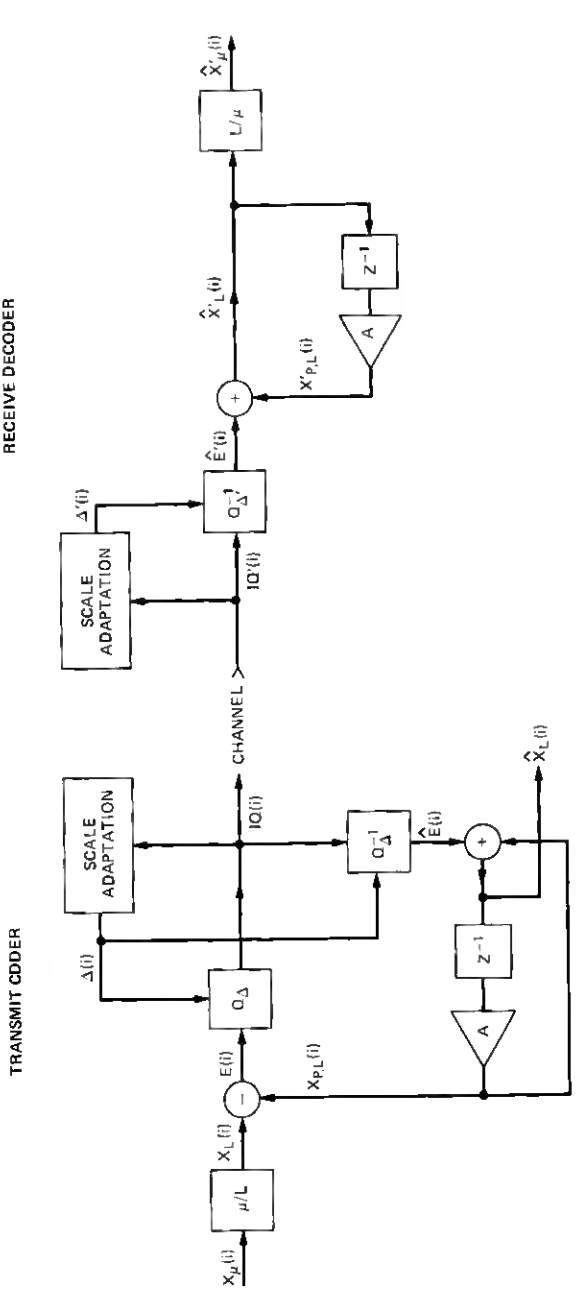

Fig. 1—Conventional ADPCM structure and interface to μ-law environment.

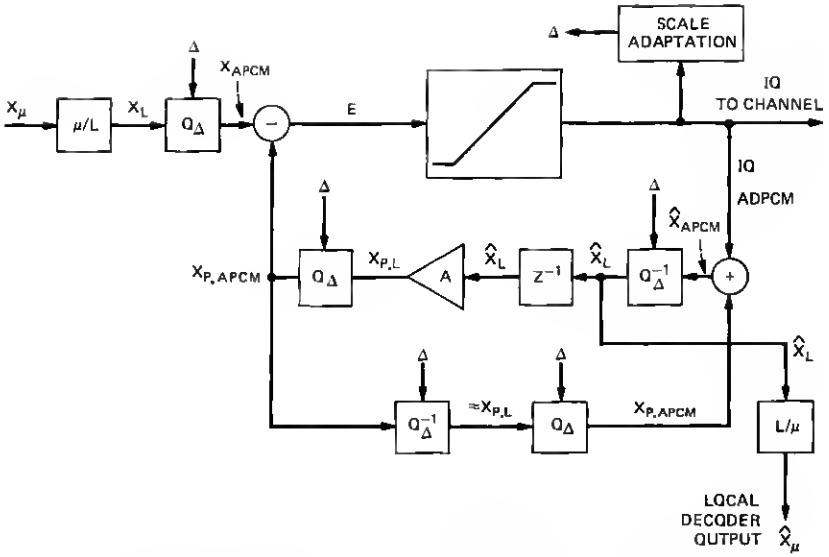

Fig. 2—Relating a conventional to a proposed ADPCM structure.

2.2 Quantization and scale adaptation details

The ROM architecture of Fig. 3 permits a very general interpretation of scale as scale index d and of Q_d and Q_d^{-1} as corresponding arbitrary quantizers. In this general framework, the adaptation rule of Refs. 1, 2, and 3 evolves to the following heuristic scale index adaptation and corresponding quantization constraint: For large (small) magnitude prediction error, the index d is increased (decreased) and the corresponding quantization Q_d and Q_d^{-1} is made coarser (finer). The quantization we propose is a marriage of uniform and μ-255 quantization; we call it semiuniform quantization. The semiuniform quantization and corresponding adaptation strategy is best explained by starting with uniform quantization where scale is synonymous to step size of the uniform quantizer.

Denoting the uniform quantizer step by Δ, the adaptation algorithm is

$$\Delta(i+1) = \Delta(i)M(i), \tag{1}$$

where $M(i) \triangleq M(|IQ(i)|) \triangleq M_{|IQ(i)|}$ are positive and have the ordering

$$M_0 \leq M_1 \leq, \cdots, \leq M_{2^{B-1}-1},$$

where

$$M_0 < 1 \text{ and } M_{2^{B-1}-1} > 1. \tag{2}$$

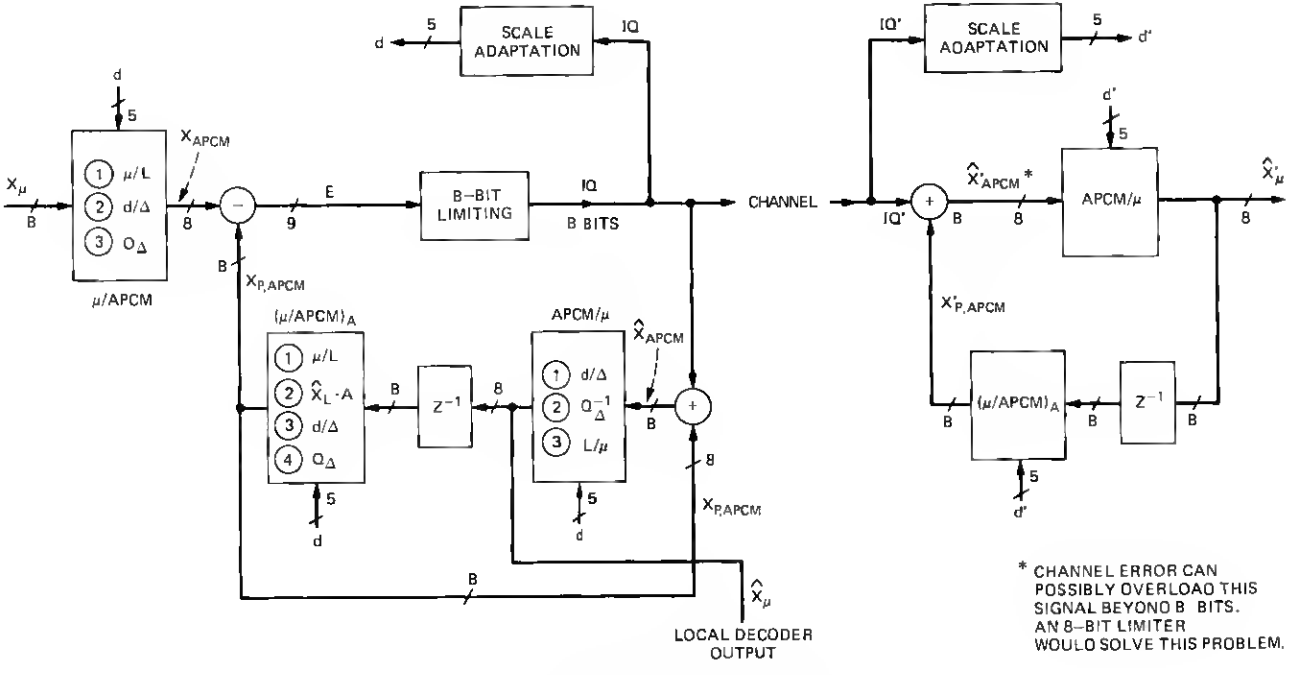

Fig. 3—The proposed ADPCM architecture.

In Refs. 2 and 5, optimized adaptation vectors

$$\vec{M} = (M_0, M_1, \cdots, M_{2^{B-1}-1})$$

are proposed and related to quantizer input variance estimation. It will be convenient to work with the log transformed version of (1)

$$d(i+1) = d(i) + m(i), \tag{3}$$

where $\Delta = \Delta_0 q^d$ and $M = q^m$ for some convenient fixed Δ_0 and base q. The scale adaptation can thus be done with the simpler iteration (3), while the translation from d to Δ (d/Δ) is done in the ROM generating software, as indicated in Fig. 3. We next discuss the finite precision aspects of a digital realization and the associated ROM size implications.

Consider first the scale adaptation algorithm. We assume d to be a K-bit nonnegative integer, $0 \le d \le 2^K - 1$, representing 2^K permissible scales. Thus, $\Delta_{\min} = \Delta_0$ and $\Delta_{\max} = \Delta_0 q^{(2^K-1)}$. To permit the ADPCM coder to approximate the idle channel performance of the μ-law coder, we let the minimum step size Δ_0 equal the first chord μ-law step size which, for the 14-bit linear representation we use, corresponds to $\Delta_0 = 2$. To accommodate the 40- to 50-dB dynamic range of a speech signal, it is desired that

$$\frac{\Delta_{\max}}{\Delta_{\min}} = q^{(2^K-1)} \tag{4}$$

be between 100 and 300. The step-size resolution factor q should not be greater than 2, which is the μ-law resolution; nor is much to be gained by making q too close to 1. A useful design range is $q \in [2^{1/4}, 2]$ and $K \in [3, 5]$ with a smaller q requiring the larger K. Figure 3 assumes $K = 5$, although a smaller value might be sufficient. Figure 4 is a block diagram for the scale adaptation. We show a scale adaptation preprocessor where the adaptation increment index IA is based on the quantizer output past history such as min-max, averaging, or other.* For concreteness, we assume IA to be the three most significant magnitude bits of IQ. We have introduced F additional fractional bits for the internal scale index which we designate by d_I. The scale adaptation iteration is

$$d_I(i+1) = \begin{cases} 0 & \text{if} \quad d_I(i) + m(i) < 0 \\ d_I(i) + m(i) & \text{otherwise} \\ d_{\max} & \text{if} \quad d_I(i) + m(i) > d_{\max}, \end{cases} \tag{5}$$

* In the hardware realization we have found useful a two-word memory preprocessor where the most significant previous $|IQ|$ bit in addition to the two current most significant $|IQ|$ bits determines the current IA.

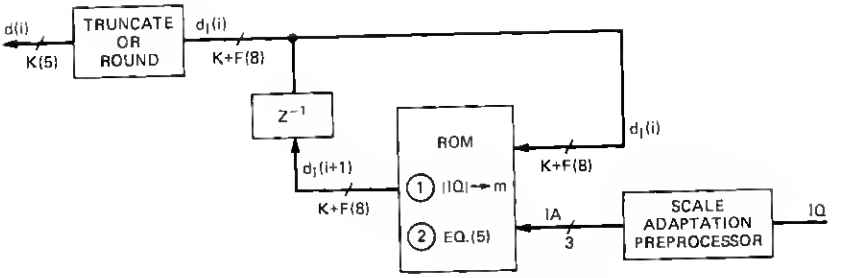

Fig. 4—Scale adaptation block diagram.

where d_I and $m(i)$ have K-integer plus F-fractional-bit representation and $d_{\max} \leq \log_q(16{,}000/2^{B-1})$ (where the upper bound on $d_{\max}$ corresponds to the step size necessary to accommodate the largest possible prediction error (16,000) without overload).* The F fractional bits are useful for two reasons. First, they permit adaptation design flexibility even with $q = 2$; for example, with $q = 2$ and $F = 0$, the largest step-reducing multiplier (closest to l) is $2^{-1} = 0.5$ while with $F = 3$ the largest step-reducing multiplier is $2^{-0.125} \simeq 0.9$. Second, they are useful in the control of adaptation mistrack due to channel errors.[6,7] Figure 4 assumes $K = 5$, $F = 3$, and up to 8 distinct adaptation multipliers implying a $2^{11} \times 8$ ROM.

Consider next the quantization. For strictly uniform quantization, large amplitude, slowly varying ADPCM input signals would generate small step sizes and large magnitudes for X_{APCM}, $\hat{X}_{\text{APCM}}$, and $X_{P,\text{APCM}}$. These large magnitudes might require a large number of bits to avoid amplitude overload. The amplitude overload problem as a function of the number of bits/word ($\leq$14) must be assessed. We propose to avoid the amplitude overload problem with the following semiuniform quantization scheme. For each scale index d, there is a threshold value $T(d)$ such that for $X_L < T(d)$ the quantization is uniform, while for $X_L > T(d)$ the quantization is a copy of the μ-law quantizer as illustrated in Fig. 5. The threshold $T(d)$ is located at approximately a μ-law segment boundary such that, for the segments below $T(d)$, the segment step sizes are $\leq \Delta = \Delta_0 q^d$ while, for the segments above $T(d)$, the segment step sizes are $> \Delta$. Thus wasteful quantizer level assignments (and therefore bits) are avoided when the final μ-law quantization would ignore them anyway. Figure 6 shows the details of uniform and semiuniform quantization assuming $\Delta = 3$. The important feature of the semiuniform quantization is that the number of quantizing inter-

* For example, for $q = \sqrt{2}$ and $B = 4$, $d_{\max} \simeq 22$ and therefore $K = 4$ with $d_{\max} = 15$ or $K = 5$ with $d_{\max} \epsilon [16, 22]$ should be satisfactory.

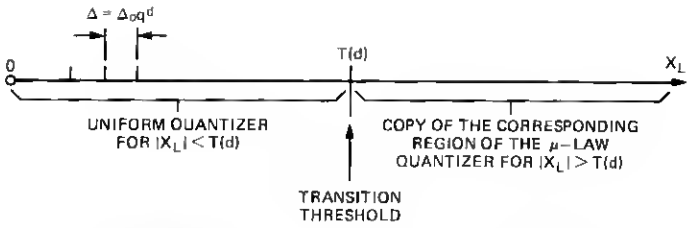

Fig. 5—Basic structure of semiuniform quantizer (positive half).

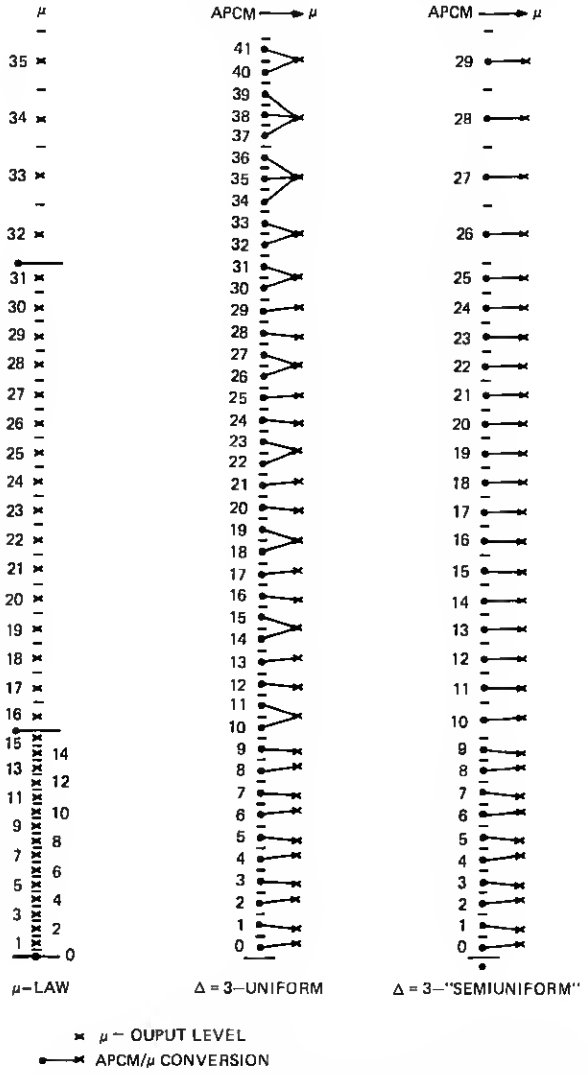

Fig. 6—Illustrating uniform and semiuniform quantization.

vals is ≤256 and therefore 8 bits is sufficient for the representation of X_{APCM}, $\hat{X}_{\text{APCM}}$, and $X_{P,\text{APCM}}$. The blocks μ/APCM, APCM/μ, and $(\mu/\text{APCM})_A$ including a sign magnitude/2's-complement or inverse-conversion can each be realized with a $2^{13} \times 8$ ROM, or alternatively one can use $2^{12} \times 7$ ROMs with the sign bit feeding around the ROMs to separate sign magnitude/2's-complement converters.

Although the optimizations reported in Refs. 2, 3, and 5 do not quantitatively apply to our ADPCM coder, the qualitative concepts developed there with respect to loading, adaptation speed, and others are very useful in tuning our coder, and they could be utilized in our hardware realization.

The next section briefly discusses two approaches to dealing with transmitter-receiver scale mistrack due to channel errors. The discussion serves to illustrate the versatility of our ROM architecture.

III. CHANNEL ERROR AND SCALE MISTRACK

In a backward adapting quantizer, the possibility of scale mistrack due to channel errors must be considered. A robust quantizer is obtained by modifying (3) [and similarly modifying (5)] to

$$d(i+1) = \beta d(i) + m(i), \tag{6}$$

where β is some number less than but close to 1. The modification in (6) is incorporated into our architecture by a change of the scale adaptation ROM. References 6 and 7 discuss in detail performance aspects and design guidelines related to the finite arithmetic implementation of (6). Also, the improved quantizer reconstruction strategy recommended for the robust quantizer in Ref. 8 can be implemented by a change in the APCM/μ reconstruction table, i.e., the APCM/μ ROM. These improvements require no additional hardware.

An alternate approach to scale mistrack control is to modify (3) to

$$d_I(i+1) = d_I(i) + m(i, d_I(i)), \tag{7a}$$

where

$$m(i, d_I(i)) = \begin{cases} m(i) - 2^{-F} & D_2 < d_I(i) \\ m(i) & D_1 \le d_I(i) < D_2 \\ m(i) + 2^F & d_I(i) < D_1, \end{cases} \tag{7b}$$

$\vec{m} = \log_q \vec{M}$ is an optimized adaptation vector in the absence of channel errors, and D_1 and D_2 (e.g., $D_1 = (1/3)d_{\max}$ and $D_2 = (2/3)d_{\max}$) are design constants which, for the reason to follow, we call mistrack correction levels. Each iteration that $d_I(i)$ and $d_I'(i)$ are on opposite sides of n ($n = 1$ or 2) of the correction levels, the mistrack $|d_I(i) -$

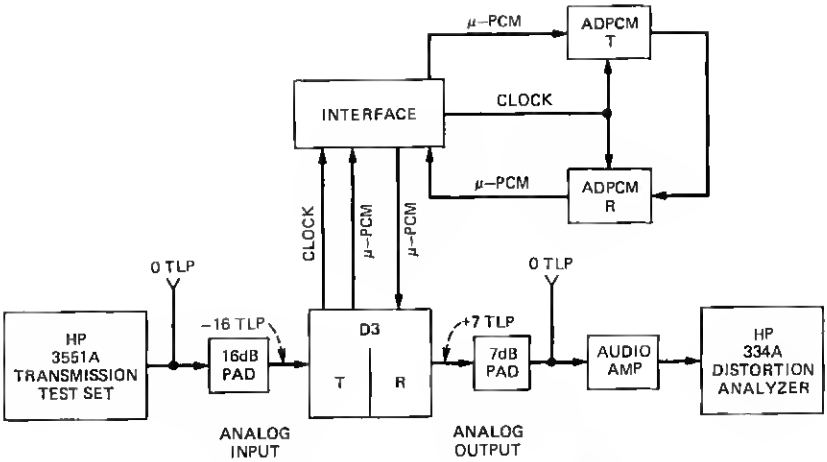

Fig. 7—ADPCM coder performance measurements.

$d_i(i)|$ is reduced by $n2^{-F}$. For* $2^{-F} \ll |m_{\min}|$ the approach in (7) should avoid the error-free dynamic range penalty[6,7] associated with (6). Again, the important point is that (7) (or a generalization of it to an arbitrary number of correcting levels) can be incorporated into the codec by a simple ROM table modification without hardware or complexity penalty.

IV. HARDWARE REALIZATION AND PERFORMANCE

We have constructed an ADPCM coder which interfaces directly with a *D*3 bank. Because our coder uses parallel processing and the ROMs are relatively fast, we can share the coder among all 24 voice trunks of the channel bank. In fact, over 120 trunks could be accommodated if we so desire. The coder is implemented with three wire-wrapped circuit packs: one each for the interface, the transmitter, and the receiver.

For the signal-to-noise measurements, we chose a prediction coefficient $A = 0.8$ and a scale adaptation vector of $(-1, 0, 0, 0, 1, 2, 3, 5)$ with a base $q = \sqrt{2}$. Because of the flexibility of the coder, we were able to vary the number of bits per sample feeding the transmission channel as well as the local decoder. The measurement arrangement is shown in Fig. 7. The signal-to-noise versus frequency for the coder is shown by the graph of Fig. 8. The measurements were made with the coder operating in the 5-bit ADPCM mode and with 5 bits in the

* If it is desired to have a small m such as $m = \pm 2^{-F}$ or $m = 0$, then (7b) can be simply modified to exclude the perturbation 2^{-F} for those m. The ROM modification is again straightforward. Also, interleaving as in Ref. 6 should reduce the required F.

feedback circuit. The analog input was −15 dBm0. As expected, the coder performance drops with increased frequency. Above 1500 Hz when the adjacent sample correlation is below 0.4, the differential mode of coding has no advantage and in fact has penalty. The remainder of the performance tests were made by measuring the signal-to-noise ratio versus amplitude for 1005 Hz. These measurements were made for the following coder options:

(*i*) ADPCM with the number of transmit bits equal to 5 and the number of feedback bits equal to 5, 4, and 3.

(*ii*) ADPCM with the number of transmit and feedback bits both equal to 4 and 3.

Figure 9 demonstrates the performance insensitivity as the number of feedback bits is dropped. Figure 10 shows the performance for 3 and

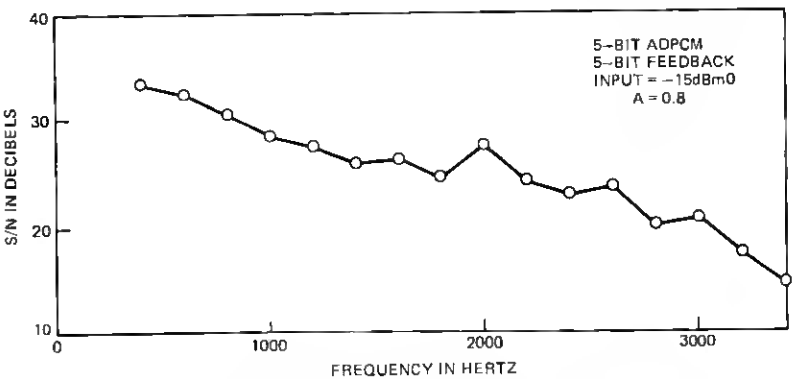

Fig. 8—ADPCM coder performance, frequency response.

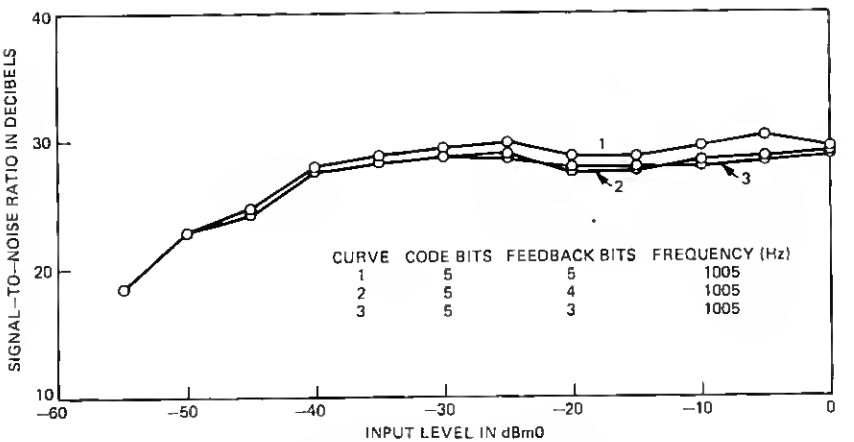

Fig. 9—ADPCM coder performance, 5-bit.

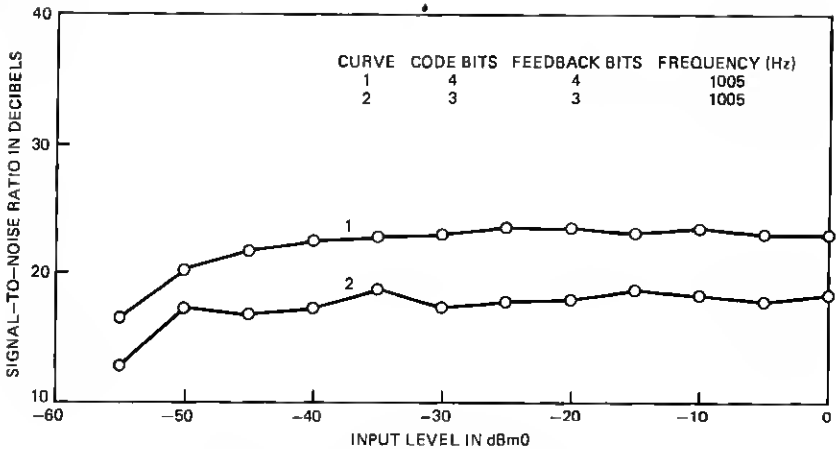

Fig. 10—ADPCM coder performance, 4- and 3-bit.

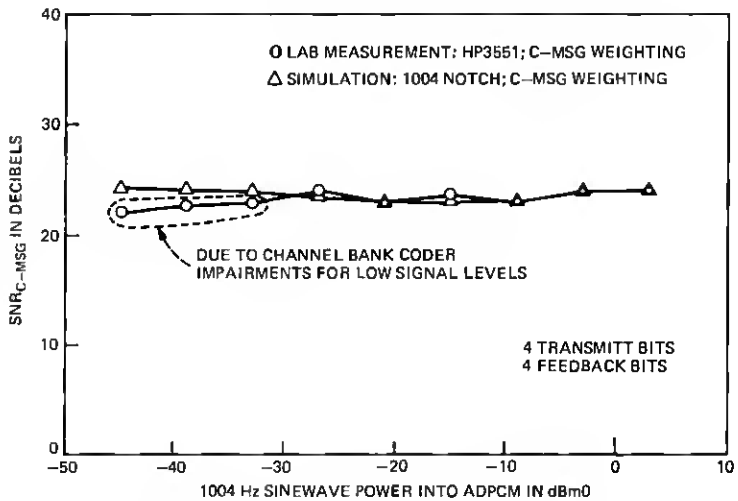

Fig. 11—Laboratory measurement vs simulation.

4 bits. Experimental and simulation results for 1004 Hz coding with 4-bit ADPCM are compared in Fig. 11.

V. ACKNOWLEDGMENT

We thank M. R. Aaron for his advice and encouragement.

REFERENCES

1. P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive Quantization in Differential PCM Coding of Speech," B.S.T.J., *52*, No. 7 (September 1973), pp. 1105–1118.

2. N. S. Jayant, "Adaptive Quantization With a One Word Memory," B.S.T.J., *52*, No. 7 (September 1973), pp. 1119–1144.
3. D. J. Goodman and A. Gersho, "Theory of an Adaptive Quantizer," IEEE Trans. Comm., *COM-22*, No. 8 (August 1974), pp. 1037–1045.
4. P. Noll, "A Comparative Study of Various Quantization Schemes for Speech Encoding," B.S.T.J., *54*, No. 9 (November 1975), pp. 1597–1614.
5. B. McDermott, C. Scagliola, and D. Goodman, "Perceptual and Objective Evaluation of Speech Processed by Adaptive Differential PCM," B.S.T.J., *57*, No. 5 (May–June 1978), pp. 1597–1618.
6. D. Mitra and B. Gotz, "An Adaptive PCM System Designed for Noisy Channels and Digital Implementations," B.S.T.J., *57*, No. 7 (September 1978), pp. 2727–2763.
7. D. J. Goodman and C. Scagliola, "Channel Error Recovery in Digital Realizations of the Robust Adaptive Quantizer," to be published at IEEE Trans. on Acoustics, Speech and Signal Processing.
8. D. Mitra, "A Generalized Adaptive Quantization System With a New Reconstruction Method for Noisy Transmission," International Conference on Acoustics, Speech and Signal Processing, Washington, April 1979.